

Web & N-Tier

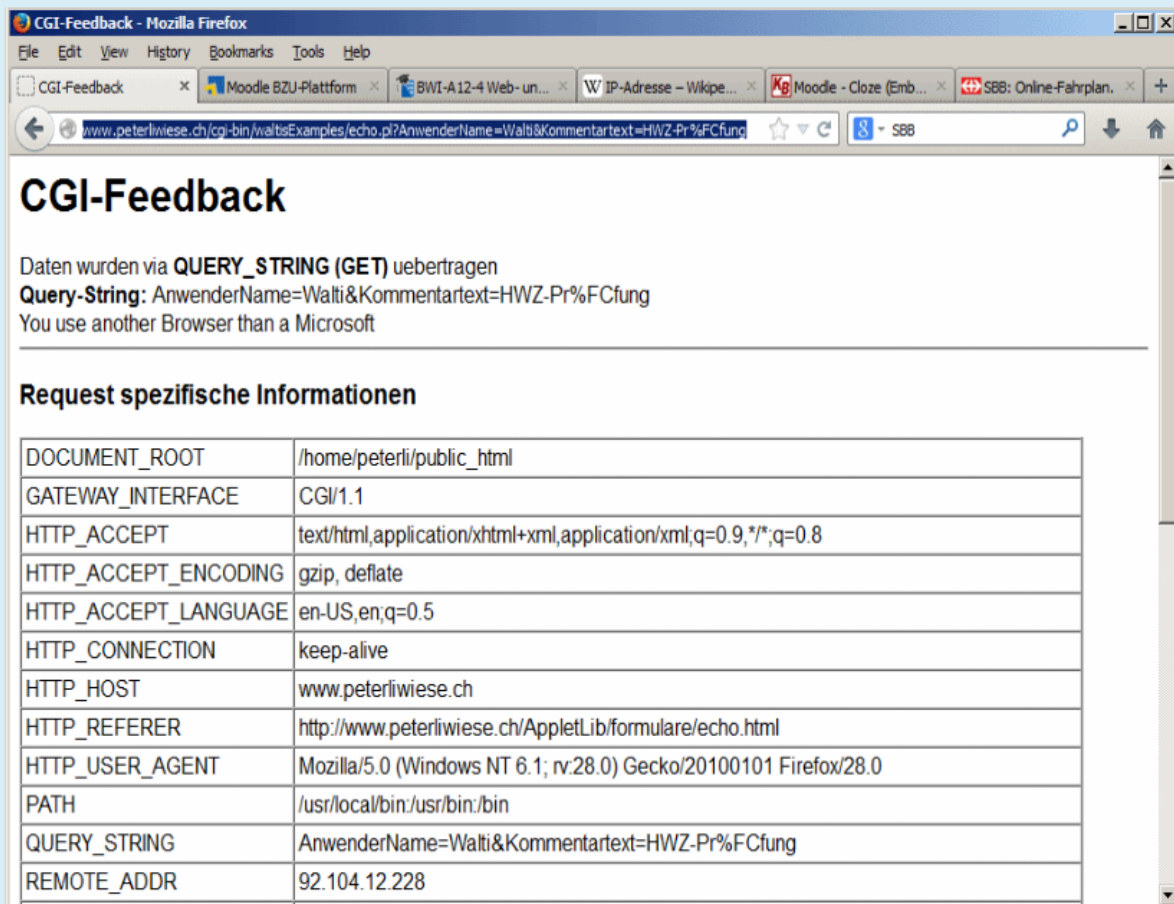
Last Updated: 20.5.21

Large HTML-Formulare

Ich habe folgendes Formular definiert:

```
<form action="/cgi-bin/waltisExamples/echo.pl" method=get>
  Name: <input size=40 maxlength=40 name="xxxxxxxxxx" Value="Walti">
  Text: <textarea rows=5 cols=30 name="yyyyyyyyyyyy" wrap=
virtual>HWZ_Prüfung</textarea>
  <input type=submit value="Absenden mit GET">
</form>
```

Nach dem drücken des Knopfes "Absenden mit GET" erhalte ich folgende Antwort Seite:



CGI-Feedback

Daten wurden via **QUERY_STRING (GET)** uebertragen
Query-String: AnwenderName=Walti&Kommentartext=HWZ-Prüfung
You use another Browser than a Microsoft

Request spezifische Informationen

DOCUMENT_ROOT	/home/peterli/public_html
GATEWAY_INTERFACE	CGI/1.1
HTTP_ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
HTTP_ACCEPT_ENCODING	gzip, deflate
HTTP_ACCEPT_LANGUAGE	en-US,en;q=0.5
HTTP_CONNECTION	keep-alive
HTTP_HOST	www.peterliwiese.ch
HTTP_REFERER	http://www.peterliwiese.ch/AppletLib/formulare/echo.html
HTTP_USER_AGENT	Mozilla/5.0 (Windows NT 6.1; rv:28.0) Gecko/20100101 Firefox/28.0
PATH	/usr/local/bin:/usr/bin:/bin
QUERY_STRING	AnwenderName=Walti&Kommentartext=HWZ-Prüfung
REMOTE_ADDR	92.104.12.228

Welchen Namen haben die Eingabe-Felder im Formular?

xxxxxxxxxx

yyyyyyyyyyyy

Vorschau Frage HTML-Formulare

Frage 1

Antwort
gespeichert

Ich habe folgendes Formular definiert:

```
<form action="/cgi-bin/waltisExamples/echo.pl" method=get>
Name: <input size=40 maxlength=40 name="xxxxxxxx" Value="Walti">
Text: <textarea rows=5 cols=30 name="yyyyyyyyyy" wrap=
virtual>HWZ_Prüfung</textarea>
<input type=submit value="Absenden mit GET">
</form>
```

Nach dem drücken des Knopfes "Absenden mit GET" erhalte ich folgende Antwort Seite:

CGI-Feedback

Daten wurden via **QUERY_STRING (GET)** uebertragen
Query-String: AnwenderName=Walti&Kommentartext=HWZ-Prüfung
You use another Browser than a Microsoft

Request spezifische Informationen

DOCUMENT_ROOT	/home/peterli/public_html
GATEWAY_INTERFACE	CGI/1.1
HTTP_ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
HTTP_ACCEPT_ENCODING	gzip, deflate
HTTP_ACCEPT_LANGUAGE	en-US,en;q=0.5
HTTP_CONNECTION	keep-alive
HTTP_HOST	www.peterliwiese.ch
HTTP_REFERER	http://www.peterliwiese.ch/AppletLib/formulare/echo.html
HTTP_USER_AGENT	Mozilla/5.0 (Windows NT 6.1; rv:28.0) Gecko/20100101 Firefox/28.0
PATH	/usr/local/bin:/usr/bin:/bin
QUERY_STRING	AnwenderName=Walti&Kommentartext=HWZ-Prüfung
REMOTE_ADDR	92.104.12.228

Welchen Namen haben die Eingabe-Felder im Formular?

xxxxxxxx AnwenderName

yyyyyyyyyy Kommentartext

ge Content - Layout

Content von Layout zu trennen ist aus Maintenance Gründen sehr wichtig.

Bei wird Java-Code durch viele Aufrufe von String-Operationen mit HTML Code schwer leserlich gemacht. Auf der anderen Seite wird in HTML-Code durch Java-Code erweitert.

JSP
Servlet
CGI
JS

Servlet
JSP
CGI
JS

ge Content - Layout

Content von Layout zu trennen ist aus Maintenance Gründen sehr wichtig.

Bei wird Java-Code durch viele Aufrufe von String-Operationen mit HTML Code

schwer leserlich gemacht. Auf der anderen Seite wird in HTML-Code durch Java-Code erweitert.

ge EJB

Ein EJB () ist der Ueberbegriff für serverseitige Komponenten. Diese

werden unterteilt in sowie , welche

weiter unterteilt werden in **Stateful** und **Stateless**.

Der Application-Server kann Session Beans poolen und für mehrere Requests von verschiedenen Sessions wieder verwenden.

Elemtry Java Bean
Enterprise Java Bean

Entity Beans
Session Bean
Message Driven Bean

Entity Beans
Session Bean
Message Driven Bean

ge EJB

Ein EJB () ist der Ueberbegriff für serverseitige Komponenten. Diese werden unterteilt in sowie , welche weiter unterteilt werden in **Stateful** und **Stateless**.

Der Application-Server kann Session Beans poolen und für mehrere Requests von verschiedenen Sessions wieder verwenden.

ge Servlet





Bewerten sie folgende Aussagen:

- Welches ist die Basis-Technologie in Java um Server-Seitige Programme zu schreiben?
- Bei einem Request auf ein Servlet wird jedesmal vom Web-Server ein neuer Process auf dem Server gestartet.
- Ein JSP wird beim Deployment oder vor dem ersten Request in ein Servlet umgewandelt.
- Was braucht ein Servlet neben dem Web-Server noch, um auf Requests reagieren zu können?

JSP JSF J2EE Servlet TPC CGI Perl	Stimmt Ist nicht definiert Stimmt nicht	Stimmt nicht Stimmt Ist gerade umgekehrt Hat nichts miteinander zu tun	cgi-Server Application-Server DB-Server
---	---	---	---

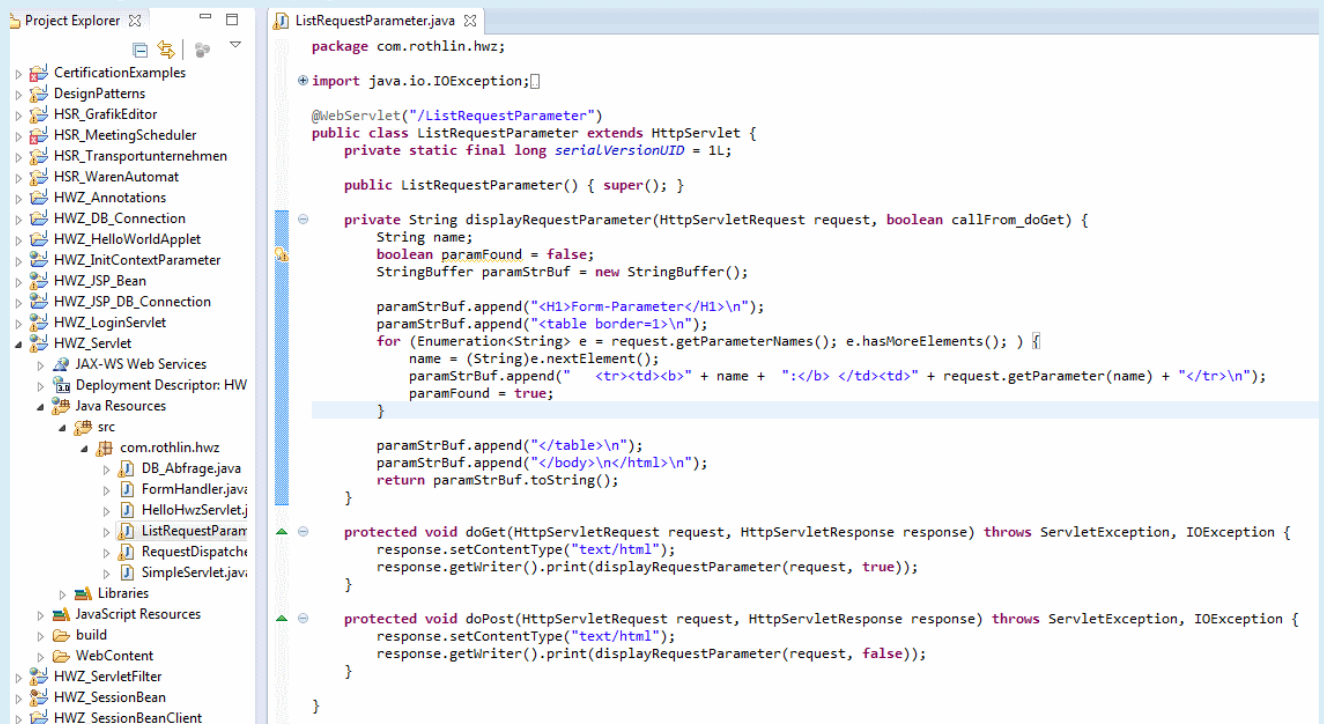
ge Servlet

Bewerten sie folgende Aussagen:

- Welches ist die Basis-Technologie in Java um Server-Seitige Programme zu schreiben?
 
- Bei einem Request auf ein Servlet wird jedesmal vom Web-Server ein neuer Process auf dem Server gestartet. 
- Ein JSP wird beim Deployment oder vor dem ersten Request in ein Servlet umgewandelt.
 
- Was braucht ein Servlet neben dem Web-Server noch, um auf Requests reagieren zu können? 

ge Servlet Call

Sie haben folgendes Servlet in Eclipse geöffnet:



The screenshot shows the Eclipse IDE with a project named 'HWZ_Servlet' in the Project Explorer. The source code of the 'ListRequestParameter.java' file is displayed in the main editor. The code is a Java Servlet that extends 'HttpServlet' and implements the 'doGet' and 'doPost' methods. It uses a 'StringBuffer' to build an HTML response that lists the request parameters in a table. The 'doGet' method calls 'displayRequestParameter' with 'true', while 'doPost' calls it with 'false'.

```
package com.rothlin.hwz;

import java.io.IOException;

@WebServlet("/ListRequestParameter")
public class ListRequestParameter extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public ListRequestParameter() { super(); }

    private String displayRequestParameter(HttpServletRequest request, boolean callFrom_doGet) {
        String name;
        boolean paramFound = false;
        StringBuffer paramStrBuf = new StringBuffer();

        paramStrBuf.append("<H1>Form-Parameter</H1>\n");
        paramStrBuf.append("<table border=1>\n");
        for (Enumeration<String> e = request.getParameterNames(); e.hasMoreElements(); ) {
            name = (String)e.nextElement();
            paramStrBuf.append("    <tr><td><b>" + name + " :</b> </td><td>" + request.getParameter(name) + "</tr>\n");
            paramFound = true;
        }

        paramStrBuf.append("</table>\n");
        paramStrBuf.append("</body>\n</html>\n");
        return paramStrBuf.toString();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.setContentType("text/html");
        response.getWriter().print(displayRequestParameter(request, true));
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.setContentType("text/html");
        response.getWriter().print(displayRequestParameter(request, false));
    }
}
```

Auf dem Web-Browser sehen Sie folgende Antwort:

Form-Parameter

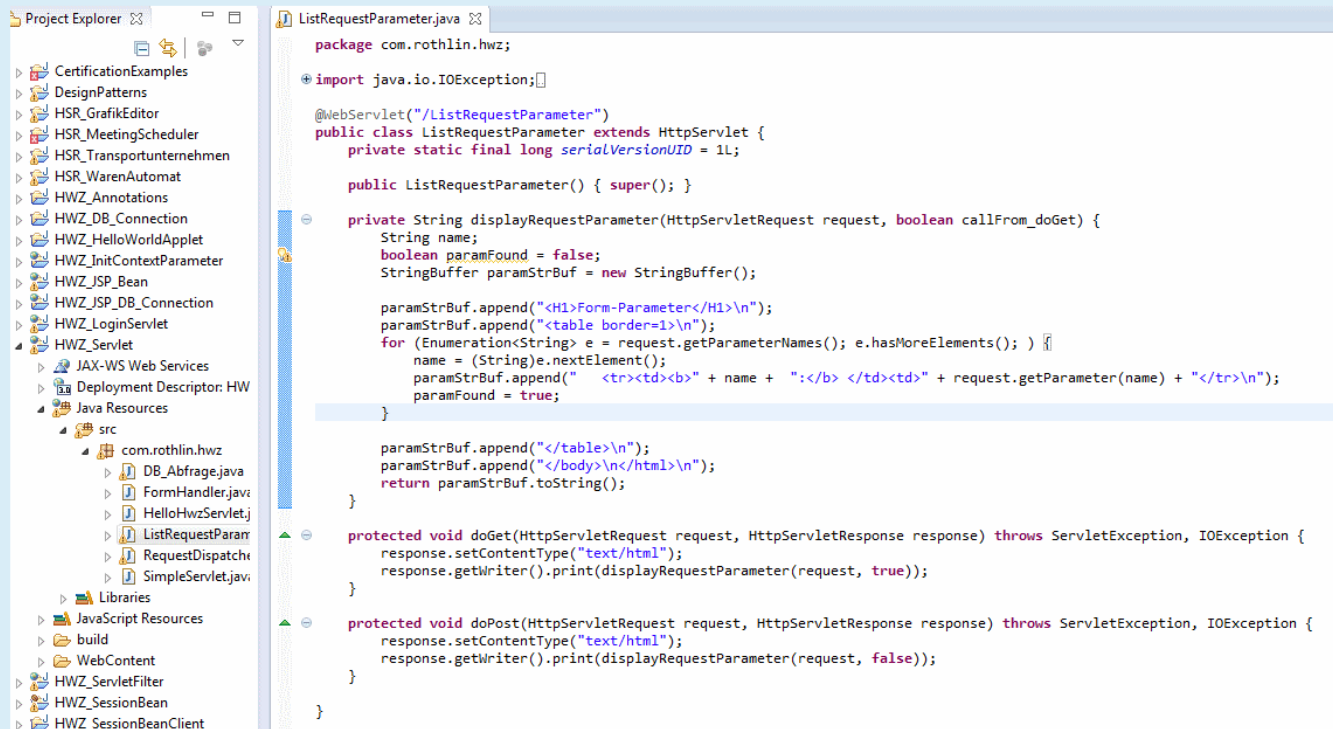
Firma:	HWZ
Anlass:	Ringvorlesung

Wie sieht die vollständige URL aus, welche zu diesem Output geführt hat, falls die Parameter mit der Methode GET übertragen wurden?

http://localhost:8080/HWZ_Servlet/

ge Servlet Call

Sie haben folgendes Servlet in Eclipse geöffnet:



```
package com.rothlin.hwz;

import java.io.IOException;

@WebServlet("/ListRequestParameter")
public class ListRequestParameter extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public ListRequestParameter() { super(); }

    private String displayRequestParameter(HttpServletRequest request, boolean callFrom_doGet) {
        String name;
        boolean paramFound = false;
        StringBuffer paramStrBuf = new StringBuffer();

        paramStrBuf.append("<H1>Form-Parameter</H1>\n");
        paramStrBuf.append("<table border=1>\n");
        for (Enumeration<String> e = request.getParameterNames(); e.hasMoreElements(); ) {
            name = (String)e.nextElement();
            paramStrBuf.append("  <tr><td><b>" + name + " :</b> </td><td>" + request.getParameter(name) + "</tr>\n");
            paramFound = true;
        }

        paramStrBuf.append("</table>\n");
        paramStrBuf.append("</body>\n</html>\n");
        return paramStrBuf.toString();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.setContentType("text/html");
        response.getWriter().print(displayRequestParameter(request, true));
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.setContentType("text/html");
        response.getWriter().print(displayRequestParameter(request, false));
    }
}
```

Auf dem Web-Browser sehen Sie folgende Antwort:

Form-Parameter

Firma:	HWZ
Anlass:	Ringvorlesung

Wie sieht die vollständige URL aus, welche zu diesem Output geführt hat, falls die Parameter mit der Methode GET übertragen wurden?

http://localhost:8080/HWZ_Servlet/ListRequestParameter?Firma=HWZ&Anlass=Ringvorlesung

Wie Servlet calls EJB Methods

Von einem Entwickler eines EJB bekommen Sie folgende zwei Java-Files:

CalculatorInterface_1.java:

```
package com.rothlin.hwz;
import javax.ejb.Local;

@Local
public interface CalculatorInterface_1 {
    public int multInt(int a, int b);
    public int divInt(int a, int b);
    public double getSteigung(double x1, double y1, double x2, double y2);
    public double getY_Schnittpunkt(double x1, double y1, double x2, double y2);
}
```

CalculatorInterface_2.java:

```
package com.rothlin.hwz;
import javax.ejb.Local;

@Remote
public interface CalculatorInterface_2 {
    public int multInt(int a, int b);
    public int divInt(int a, int b);
    public boolean isPrimzahl(int zahl);
}
```

Sie schreiben nun in Ihrer eigenen Web-Applikation ein Servlet. Auf welche Methoden können Sie zugreifen:

multInt: ☐

divInt: ☐

getSteigung: ☐

getY_Schnittpunkt: ☐

isPrimezahl: ☐

Zugriff nicht möglich
Zugriff möglich

Zugriff nicht möglich
Zugriff möglich

Zugriff nicht möglich
%0%Zugriff möglich

Zugriff nicht möglich
%0%Zugriff möglich

Zugriff nicht möglich
Zugriff möglich

ge Servlet calls EJB Methods

Von einem Entwickler eines EJB bekommen Sie folgende zwei Java-Files:

CalculatorInterface_1.java:

```
package com.rothlin.hwz;
import javax.ejb.Local;

@Local
public interface CalculatorInterface_1 {
    public int multInt(int a, int b);
    public int divInt(int a, int b);
    public double getSteigung(double x1, double y1, double x2, double y2);
    public double getY_Schnittpunkt(double x1, double y1, double x2, double y2);
}
```

CalculatorInterface_2.java:

```
package com.rothlin.hwz;
import javax.ejb.Local;

@Remote
public interface CalculatorInterface_2 {
    public int multInt(int a, int b);
    public int divInt(int a, int b);
    public boolean isPrimzahl(int zahl);
}
```

Sie schreiben nun in Ihrer eigenen Web-Applikation ein Servlet. Auf welche Methoden können Sie zugreifen:

multInt:

divInt:

getSteigung:

getY_Schnittpunkt:

isPrimezahl:

Wie Servlet calls EJB Methods (Annotation)

Sie schreiben ein Servlet innerhalb einer bestehenden Web-Applikation. Diese Web-Applikation hat bereits ein EJB wie folgt implementiert.:

CalculatorInterface_2.java:

```
@Remote
public interface CalculatorInterface_2 {
    public int multInt(int a, int b);
    public int divInt(int a, int b);
    public boolean isPrimzahl(int zahl);
}
```

CalculatorInterface_1.java:

```
@Local
public interface CalculatorInterface_1 {
    public int multInt(int a, int b);
    public int divInt(int a, int b);
    public double getSteigung(double x1, double y1, double x2, double y2);
    public double getY_Schnittpunkt(double x1, double y1, double x2, double y2);
}
```

Calculator.java:

```
@Stateless
public class Calculator implements CalculatorInterface_2, CalculatorInterface_1 { .....
```

Sie implementieren nun Ihr eigenes Servlet MyCalcApp.java. Wie kommen Sie zur richtige Objekt-Referenz und wie rufen Sie die Methode multInt() des EJB auf:

MyCalcApp.java:

```
@WebServlet("/MyCalcApp")
public class MyCalcApp extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public MyCalcApp() { super(); }

    private void serviceRequest(HttpServletRequest request, HttpServletResponse response, boolean callFrom_doGet) throws ServletException, IOException {

        StringBuffer sb = new StringBuffer();
        sb.append("5*6=" + res + "\n");
        response.setContentType("text/html");
        response.getWriter().print(sb);
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        serviceRequest(request, response, true);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        serviceRequest(request, response, false);
    }
}
```

```
@EJB CalculatorInterface_2 calculator
@EJB CalculatorInterface_1 calc = new CalculatorInterface_1();
@EJB CalculatorInterface_1 calc;
Calculator calc = (CalculatorRemote) ctx.lookup("java:global/HWZ_Servlet/servletcom.rothlin.hwz.CalculatorRemote");
Calculator calc = new CalculatorInterface_1();
```

```
int res = Calculator.multInt(5,6);
int res = multInt(5,6);
int res = calc.multInt(5,6);
float res = Calculator.multInt(5,6);
float res = calculator.multInt(5,6);
```

Wie Servlet calls EJB Methods (Annotation)

Sie schreiben ein Servlet innerhalb einer bestehenden Web-Applikation. Diese Web-Applikation hat bereits ein EJB wie folgt implementiert.:

[CalculatorInterface_2.java:](#)

```
@Remote
public interface CalculatorInterface_2 {
    public int multInt(int a, int b);
    public int divInt(int a, int b);
    public boolean isPrimzahl(int zahl);
}
```

[CalculatorInterface_1.java:](#)

```
@Local
public interface CalculatorInterface_1 {
    public int multInt(int a, int b);
    public int divInt(int a, int b);
    public double getSteigung(double x1, double y1, double x2, double y2);
    public double getY_Schnittpunkt(double x1, double y1, double x2, double y2);
}
```

[Calculator.java:](#)

```
@Stateless
public class Calculator implements CalculatorInterface_2, CalculatorInterface_1 { .....
```

Sie implementieren nun Ihr eigenes Servlet MyCalcApp.java. Wie kommen Sie zur richtige Objekt-Referenz und wie rufen Sie die Methode multInt() des EJB auf:

[MyCalcApp.java:](#)

```
@WebServlet("/MyCalcApp")
public class MyCalcApp extends HttpServlet {
    private static final long serialVersionUID = 1L;

    @EJB CalculatorInterface_1 calc;

    public MyCalcApp() { super(); }

    private void serviceRequest(HttpServletRequest request, HttpServletResponse response, boolean callFrom doGet) throws ServletException, IOException {

        int res = calc.multInt(5,6);

        StringBuffer sb = new StringBuffer();
        sb.append("5*6=" + res + "\n");
        response.setContentType("text/html");
        response.getWriter().print(sb);
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        serviceRequest(request, response, true);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        serviceRequest(request, response, false);
    }
}
```

ge Servlet Details 1

Sie haben folgen Code Ausschnitt aus einem Servlet:

```
String callerAppl = request.getParameter("Application");  
if (!((callerAppl != null) && (callerAppl.equalsIgnoreCase("RequestDispatcher")))) {  
  
}
```

Was wird in **String callerAppl = request.getParameter("Application");** gemacht?

Der Wert des mit dem Namen "Application" der Variable "callerAppl" zugewiesen

Wieso braucht es beim If den ersten Vergleich (**callerAppl != null**)?

Braucht es ein **&&** oder würde ein **&** auch genügen (Mit Begründung)?

Request-Parameters
Context-Parameters
Cookies
Initial-Parameters

Es braucht diesen Vergleich nicht
Es ist falsch und sollte anstelle von != ein == sein
Das Servlet würde sonst unter gewissen Umständen abstürzen

Ja, ohne && gibt es sonst in jedem Fall ein Runtime-Fehler
Ja, ohne && gibt es trotzdem ab und zu ein Runtime-Fehler
Nein, && und & ist in Java genau das Gleiche
Nein, ein & wäre das einzig Richtige

ge Servlet Details 1

Sie haben folgen Code Ausschnitt aus einem Servlet:

```
String callerAppl = request.getParameter("Application");  
if (!((callerAppl != null) && (callerAppl.equalsIgnoreCase("RequestDispatcher")))) {  
  
}
```

Was wird in **String callerAppl = request.getParameter("Application");** gemacht?

Der Wert des mit dem Namen "Application" der Variable "callerAppl" zugewiesen

Wieso braucht es beim if den ersten Vergleich (**callerAppl != null**)?

Braucht es ein **&&** oder würde ein **&** auch genügen (Mit Begründung)?

ge Server Side Script

Wie wird eine Funktion (script, servlet,...) vom Client aus auf dem Server aktiviert (aufgerufen)?

Nur eine Antwort ist korrekt.

- ☐ a. Mehrere von diesen Antworten sind möglich
- ☐ b. Remote Procedure Call (RPC)
- ☐ c. Keines von allen
- ☐ d. Remote Methode Invocation (RMI)
- ☐ e. Durch einen URL-Request

ge Server Side Script

Wie wird eine Funktion (script, servlet,...) vom Client aus auf dem Server aktiviert (aufgerufen)?

Nur eine Antwort ist korrekt.

- ☐ a. Mehrere von diesen Antworten sind möglich
- ☐ b. Keines von allen
- ☒ c. Durch einen URL-Request
- ☐ d. Remote Methode Invocation (RMI)
- ☐ e. Remote Procedure Call (RPC)

ge Servlet is called

Sie haben mit Ihrem Team ein Web-Formular (OrderForm.html) und ein Servlet (TakeOrder) entwickelt. Beides wurde erfolgreich getestet und in Produktion gebracht.

Ein anderes Team hat ebenfalls ein Web-Form (DeliveryForm.html) und möchte 1:1 das TakeOrder Servlet verwenden. Als Verantwortlicher vom TakeOrder Servlet müssen Sie zwingend folgende Details dem Entwickler des neuen Formulars weiter geben:

- Den Mimetype des vom Servlet zurück gesendeten Byte-Stream
- Die URL-des Formulars
- Die URL-des Servlets
- Die zwingend erforderlichen Namen der Formular Elemente, welche ans Servlet gesendet werden müssen
- Die Einschränkungen der möglichen Eingabewerte ins Formular
- Der benötigte Java-Script Code für die Usereingabe-Überprüfung
- Wie das Methode-Attribute im Formular gesetzt werden muss
- Welche HTML-Version für die Implementation des Formulars verwendet werden muss.

ge Servlet is called

Sie haben mit Ihrem Team ein Web-Formular (OrderForm.html) und ein Servlet (TakeOrder) entwickelt. Beides wurde erfolgreich getestet und in Produktion gebracht.

Ein anderes Team hat ebenfalls ein Web-Form (DeliveryForm.html) und möchte 1:1 das TakeOrder Servlet verwenden. Als Verantwortlicher vom TakeOrder Servlet müssen Sie zwingend folgende Details dem Entwickler des neuen Formulars weiter geben:

- Den Mimetype des vom Servlet zurück gesendeten Byte-Stream
- Die URL-des Formulars
- Die URL-des Servlets
- Die zwingend erforderlichen Namen der Formular Elemente, welche ans Servlet gesendet werden müssen
- Die Einschränkungen der möglichen Eingabewerte ins Formular
- Der benötigte Java-Script Code für die Usereingabe-Überprüfung
- Wie das Methode-Attribute im Formular gesetzt werden muss
- Welche HTML-Version für die Implementation des Formulars verwendet werden muss.

ge JavaScript

Welche Aussagen über JavaScript stimmen?

- Bei JavaScript ist normaler Java-Code in HTML eingebunden ☐
- JavaScript hat eine Java ähnliche Syntax ist aber nicht Objekt-Orientiert. ☐
- JavaScript ist eine alte Bezeichnung für JSP (Java Server Pages) ☐
- JavaScript wird sofort nach dem Rendern der HTML Seite ausgeführt. Nach diesem erstmaligen Run bestimmt der Benutzer wann welche Funktion aufgerufen wird indem er die Funktion auswählt. ☐
- Das JavaScript Code ausgeführt wird, muss man sogenannte Event-Handlers im HTML definieren. ☐
- Mittels JavaScript kann unter anderem eine Validierung der Benutzereingaben durchgeführt werden, bevor die Daten an den Server gesendet werden. ☐
- JavaScript wird durch die meisten Application-Servers unterstützt. ☐
- JavaScript kann in eigene Dateien ausgelagert werden und dann von mehreren HTML via URL referenziert werden. ☐

Wichtige Java-Script

Welche Aussagen über Java-Script stimmen?

- Bei Java-Script ist normaler Java-Code in HTML eingebunden. Falsch ☐
- Java-Script hat eine Java ähnliche Syntax ist aber nicht Objekt-Orientiert. Richtige ☒
- Java-Script ist eine alte Bezeichnung für JSP (Java Server Pages) Falsch ☐
- Java-Script wird sofort nach dem Rendern der HTML Seite ausgeführt. Nach diesem erstmaligen Run bestimmt der Benutzer wann welche Funktion aufgerufen wird indem er die Funktion auswählt. Falsch ☐
- Das Java-Script Code ausgeführt wird, muss man sogenannte Event-Handlers im HTML definieren. Richtige ☒
- Mittels Java-Script kann unter anderem eine Validierung der Benutzereingaben durchgeführt werden, bevor die Daten an den Server gesendet werden. Richtige ☒
- Java-Script wird durch die meisten Application-Servers unterstützt. Falsch, irrelevant für App-Servers ☐
- Java-Script kann in eigene Dateien ausgelagert werden und dann von mehreren HTML via URL referenziert werden. Richtige ☒

Formhandler-Aufgabe (Leistungsnachweis)

1. Erstellen Sie eine lokale Kopie des [HTML-Formulares](http://hwz.peterliwiese.ch/examples/FormTest_formHandler.html) (http://hwz.peterliwiese.ch/examples/FormTest_formHandler.html) unter **HWZ_yyyy_Nachname_Vorname.html** (z.B. HWZ_2019_Rothlin_Walter.html).

Ändern Sie in diesem File den Teil der Ehrenwörtlichen Erklärung auf Ihren Namen.

Das ist das einzige File, in welchem Sie Änderungen machen und nur dieses File geben Sie am Schluss ab (Hochladen auf MyHWZ).

2. Formatieren Sie es korrekt (Einrücken)
3. Ändern Sie das Dokument, zu einem korrekten XHTML Dokument
4. Ändern Sie nun das Formular zu einem Anmeldeformular mit folgenden Inhalt:

Anmeldeformular für Geschäftsanlasse

Hiermit melde ich mich für den Anlass vom

Name:

Vorname:

email:

PLZ: Ort:

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Komme in Begleitung von

keiner

keiner

einer

zwei

drei

 weiteren Person(en)

Kommentar

5. Binden Sie das CSS mit http://hwz.peterliwiese.ch/examples/CSS_HWZ_Style_01.css in Ihr Anmeldeformular ein. Der Titel wird nun **rot**, etwa wie folgt:

Anmeldeformular für Geschäftsanlasse

Hiermit melde ich mich für den Anlass vom

Name:
Vorname:
email:
PLZ: Ort:

ÖV Vergünstigung: ☐ Nichts ☐ 1/2 Preis ☒ GA

Mittagessen: ☐ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

Ich nehme meinen Hund mit!
Das Fressen für den Hund bringe ich selber mit.



6. Bei einer erfolgreichen Anmeldung bekommen Sie diese Seite als Antwort im Browser:

Provisorische Anmeldebestätigung

Vielen Dank **Felix Muster** für Ihre Anmeldung

Sie werden in den nächsten Tagen eine Bestätigung an **claudia@peterliwiese.ch** bekommen

Ihre Angaben wurden an **anmeldung@peterliwiese.ch** übermittelt!

Vielen Dank
Walter Rothlin (Reiseführer)

Ändern Sie Ihr Anmeldeformular auf das Template

http://www.hwz.peterliwiese.ch/examples/testSuccessTemplate_01.html

Wenn Sie nun das Anmeldeformular ausgefüllt absenden, werden einige Platzhalter nicht mit ihren Angaben ersetzt!

7. Passen Sie die Feldnamen in Ihrem Formular so an, dass diese bei der Antwort Seite auch angezeigt werden:

Anmeldebestätigung

Vielen Dank **Rothlin Walter** für Ihre Anmeldung!

Folgende Angaben wurden an anmeldung@peterliwiese.ch übermittelt!

Name: **Walter**
Vorname: **Rothlin**
email: **walter@rothlin.com**
PLZ: **8855 Ort: Wangen (SZ)**

ÖV Vergünstigung: **GA**
Mittagessen vegan: **on**
Ich komme in Begleitung von **2** weiteren Person(en)
Ihr Kommentar:
Ich nehme meinen Hund mit! Das Fressen für den Hund bringe ich selber mit.

Sie werden in den nächsten Tagen eine Bestätigung an **walter@rothlin.com** bekommen

Vielen Dank
Walter Rothlin (Reiseführer)

8. Die Postleitzahl wird nach dem absenden des Formulars auf der Server-Seite überprüft und ev. eine Fehlerseite angezeigt.
- Implementieren Sie einen Eventhandler, welcher während der Eingabe überprüft, ob nur Zahlen eingegeben werden und ob die Eingabe maximal 4 Zeichen lang ist.

Requirements:

- nur Zahlen annehmen
- nur 4 Zeichen annehmen
- es muss eine JavaScript Funktion erstellt verwendet
- der Wert im Eingabefeld muss realtime mässig beim Tippen überprüft und ev. korrigiert werden.

9. Fügen sie hinter den Feldern Name, Vorname und email ein Vorsicht-Symbol (z.B. <http://www.peterliwiese.ch/img/Warning.svg>) ein und definieren Sie im HTML-Header einen entsprechenden CSS-Style für die Darstellung, so dass ihr Formular nun etwa so aussieht:

Anmeldeformular für Geschäftsanlasse

Hiermit melde ich mich für den Anlass vom

Name: 

Vorname: 

email: 

PLZ: Ort:

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

10. Bauen Sie nun 3 Eventhandlers auf diese Felder, welche während dem Tippen überprüfen, ob die Feldregel erfüllt ist und ändern Sie das Symbol beim Eingabefeld (z.B. <http://www.peterliwiese.ch/img/Verified.svg>).

Die Regeln können Sie selber bestimmen. Danach sieht das Formular etwa so aus:

Anmeldeformular für Geschäftsanlasse

Hiermit melde ich mich für den Anlass vom

Name: ✓

Vorname: ⚠

email: ✓

PLZ: Ort:

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

Bewertung

- Jede Aufgabe wird mit **0**, **0.5** oder **1** Punkt bewertet.
- Die Note berechnet sich nach folgender Formel:

$$\text{Note} = 0.5 * \text{Punkte} + 1 \quad (6 \text{ Punkte ergeben eine } 4)$$

Erweiterung Formhandler-Aufgabe (Leistungsnachweis)

1. Erstellen Sie eine lokale Kopie des [HTML-Formulares](http://hwz.peterliwiese.ch/examples/FormTest_formHandler_2_Base.html) (http://hwz.peterliwiese.ch/examples/FormTest_formHandler_2_Base.html) unter **HWZ_2_yyyy_Nachname_Vorname.html** (z.B. HWZ_2_2019_Rothlin_Walter.html).

Ändern Sie in diesem File den Teil der Ehrenwörtlichen Erklärung auf Ihren Namen.

2. Erweitern Sie das Formular um ein Eingabefeld **Strasse**: mit einem einfachen Validierer (sie können den Validierer vom **Firstname** kopieren und entsprechend umbenennen). Nehmen Sie das Feld der **eMail** Adresse an den Schluss. Das Formular sollte dann etwa so aussehen:

Anmeldeformular Konzert

Hiermit melde ich mich für den Anlass vom

Name: !

Vorname: !

Strasse: !

PLZ: Ort:

email: !

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

3. Erweitern Sie die Validierung so, dass das Feld „Ort:“ mittels JavaScript und einem RegEx ebenfalls getestet wird und ein Symbol hinter dem Feld angezeigt wird.

Regeln für Gültigkeit:

- a. Beginnt mit Grossbuchstaben gefolgt von mindestens einem Kleinbuchstaben z.B. **Au**
- b. Optional gefolgt von mehrmaliger Wiederholung Gross-/Kleinbuchstaben getrennt durch einen Space (Leerschlag) z.B. **Siebneen Wangen**
- c. Optional am Schluss mit der Kant. Bezeichnung in Klammern z.B. **Wangen (SZ)**

Anmeldeformular Konzert

Hiermit melde ich mich für den Anlass vom

Name: ✓
Vorname: ✓
Strasse: ✓
PLZ: Ort: ✓
email: ✓

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

Anmelden

Eingaben löschen

4. Binden Sie zusätzlich folgende JavaScript Library ein:

http://www.peterliwiese.ch/JavaScriptModule/JS_Library.js

Analysieren Sie die Funktionen in dieser Library und verwenden diese in den nachfolgenden Aufgaben. Sie können dieses Beispiel (

<http://www.peterliwiese.ch/AppletLib/formulare/anmeldeformularWithJS.html>) zur Hilfe nehmen!

- Speichern Sie die validierte Eingabe eines Vornames in einem Cookie ab, indem Sie die Eventhandler Funktion für die Validierung erweitern.
- Definieren Sie einen `onload` eventhandler im Body-Tag z.B. `<body onload="getValuesFromCookies()">` und die entsprechende Funktion in JavaScript.
Sobald Sie nun einen gültigen Vornamen eingeben, wird dieser in einem Cookie gespeichert. Löschen Sie nun anschliessend diese Eingabe und machen einen Reload der Seite, wird der letzte gültige Wert im Eingabefeld angezeigt.
- Ergänzen Sie die Speicherung / Reload mittels Cookies auf folgende Eingabefelder: Name, Ort, PLZ, Ort und Email
- Implementieren Sie einen weiteren Button. Sobald Sie diesen Button klicken, werden alle Cookies, welche von dieser App geschrieben wurden, gelöscht!

Als Beispiel: <http://www.peterliwiese.ch/AppletLib/formulare/anmeldeformularWithJS.html>

Anmeldeformular Konzert

Hiermit melde ich mich für den Anlass vom

Name: ⚠

Vorname: ⚠

Strasse: ⚠

PLZ: Ort: ⚠

email: ⚠

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

5. Adress-Verify via AJAX

- a. Kombinieren Sie das Anmeldeformular mit der **Adresslocator** Applikation

(http://fhoch3.peterliwiese.ch/AdresseToKoordinates_04a_Complete.html).

```
</FORM>
<BR/>
<HR/>
<label class="LblSearchStatus" id="LblSearchStatus"></label>
<div class="SearchField">
  <div class="SearchBar">
    <input type="text" id="AdressPattern" placeholder="Please enter a search pattern" s
    <div class="fa fa-search"></div>
  </div>
  <div class="Divider"></div>
  <table id="ResultSetView" >
  </table>
</div>
</body>
</html>
```

Das Resultat sieht dann etwa so aus:

Anmeldeformular Konzert

Hiermit melde ich mich für den Anlass vom

Name: ⚠

Vorname: ⚠

Strasse: ⚠

PLZ: Ort: ⚠

email: ⚠

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

- b. Übernehmen Sie ebenfalls den Eventhandler auf dieses Feld, so dass Sie dort bereits Eingaben machen können und die Liste mit den gefundenen Adress-Locations angezeigt wird.


```
<hr/>
<label class="LblSearchStatus" id="LblSearchStatus"></label>
<div class="SearchField">
  <div class="SearchBar">
    <input type="text" id="AdressPattern" placeholder="Please enter a search pattern" />
    <div class="fa fa-search"></div>
  </div>
  <div class="Divider"></div>
  <table id="ResultSetView">
  </table>
</div>
</body>


<script>
document.getElementById("AdressPattern").addEventListener("keyup", textChanged);
</script>
</html>
```


Das Formular sieht nun etwa wie folgt aus:


Anmeldeformular Konzert


Hiermit melde ich mich für den Anlass vom

Name: 

Vorname: 

Strasse: 

PLZ: Ort: 

email: 

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

Records found:11

Peterliwiese 3

0	Peterliwiese 3	8855	Wangen SZ	8.888697624206543	47.19410705566406
1	Peterliwiese 30	8855	Wangen SZ	8.886726379394531	47.19510269165039
2	Peterliwiese 31	8855	Wangen SZ	8.88763323669434	47.194419860839844
3	Peterliwiese 32	8855	Wangen SZ	8.886526107788086	47.195152282714844

- c. Die Eingaben im Feld **Strasse** soll bei jedem Tastendruck in dieses Suchfeld übernommen werden (Koppeln der Eingabefelder). Implementieren Sie eine JavaScript Funktionen (z.B. **updateResultatenliste()**), welche bei jedem Tastendruck in das Strassenfeld, den eingegebenen Wert ins Suchfeld übernimmt. Diese Funktion rufen Sie am besten innerhalb der bestehenden **checkStrasse()** auf.

Anmeldeformular Konzert

Hiermit melde ich mich für den Anlass vom

Name: ⚠

Vorname: ⚠

Strasse: ✓

PLZ: Ort: ⚠

email: ⚠

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

Peterlwiese 33

- d. Nun erweitern Sie diese JS-Funktion so, dass der AJAX Call **textChanged()**; automatisch aufgerufen wird und die Resultatenliste so bei jeden Tastendruck im Starssen Eingabefeld, aktualisiert wird.

Anmeldeformular Konzert

Hiermit melde ich mich für den Anlass vom

Name: ⚠
Vorname: ⚠
Strasse: ✓
PLZ: Ort: ⚠
email: ⚠

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

Records found:48

0	Etzelstrasse 7 8707 Uetikon am See	8.690123558044434 47.269500732421875
1	Etzelstrasse 7 8038 Zürich	8.531061172485352 47.344905853271484
2	Etzelstrasse 7 8832 Wollerau	8.72368049621582 47.19493865966797

- e. Erweitern Sie die JavaScript Funktionen **updateResultatenliste()**, dass auch die PLZ und das Ort-Feld in die Suchkriterien übernommen werden.

Anmeldeformular Konzert

Hiermit melde ich mich für den Anlass vom

Name: ⚠
Vorname: ⚠
Strasse: ✓
PLZ: Ort: ✓
email: ⚠

ÖV Vergünstigung: ☒ Nichts ☐ 1/2 Preis ☐ GA

Mittagessen: ☒ Vegan

Ich komme in Begleitung von weiteren Person(en)

Kommentar

Anmelden

Eingaben löschen

Records found:8

0 Etzelstrasse 1 8855 Wangen SZ 8.896950721740723 47.18681716918945

1 Etzelstrasse 2 8855 Wangen SZ 8.896825790405273 47.18700408935547

2 Etzelstrasse 3 8855 Wangen SZ 8.89676570892334 47.1867790222168